(19) Europäisches Patentamt
European Patent Office
Office européen des brevets

(11) **EP 3 825 927 A1**

(12) **EUROPEAN PATENT APPLICATION**

(43) Date of publication:
26.05.2021 Bulletin 2021/21

(51) Int Cl.:
G06N 7/00 *(2006.01)*

(21) Application number: **19210420.6**

(22) Date of filing: **20.11.2019**

(84) Designated Contracting States:
AL AT BE BG CH CY CZ DE DK EE ES FI FR GB GR HR HU IE IS IT LI LT LU LV MC MK MT NL NO PL PT RO RS SE SI SK SM TR
Designated Extension States:
BA ME
KH MA MD TN

(71) Applicant: **Commissariat à l'énergie atomique et aux énergies alternatives**
75015 Paris (FR)

(72) Inventors:
- **DALGATY, Thomas**
  38054 GRENOBLE Cedex 09 (FR)
- **VIANELLO, Elisa**
  38054 GRENOBLE Cedex 09 (FR)

(74) Representative: **Brevalex**
56, Boulevard de l'Embouchure
B.P. 27519
31075 Toulouse Cedex 2 (FR)

(54) METHOD OF TRAINING A LOGISTIC REGRESSION CLASSIFIER WITH A RESISTIVE RANDOM ACCESS MEMORY

(57) The present invention concerns a method for training a logistic regression classifier on a dataset by using a resistive RAM as hardware accelerator, each row of the resistive RAM comprising *M* cells which can be programmed in a first resistance state or a second resistance state. The probability of a data element **x** belonging to a class is modelled by a logistic function applied to a score $\mathbf{w}^T\mathbf{x}$ of said element, where **w** is a parameter vector of the model. The logistic regression classifier is trained by populating the resistive RAM with samples of a model parameter vector which are obtained by MCMC sampling. Once populated, the resistive RAM can be used for classifying new data.

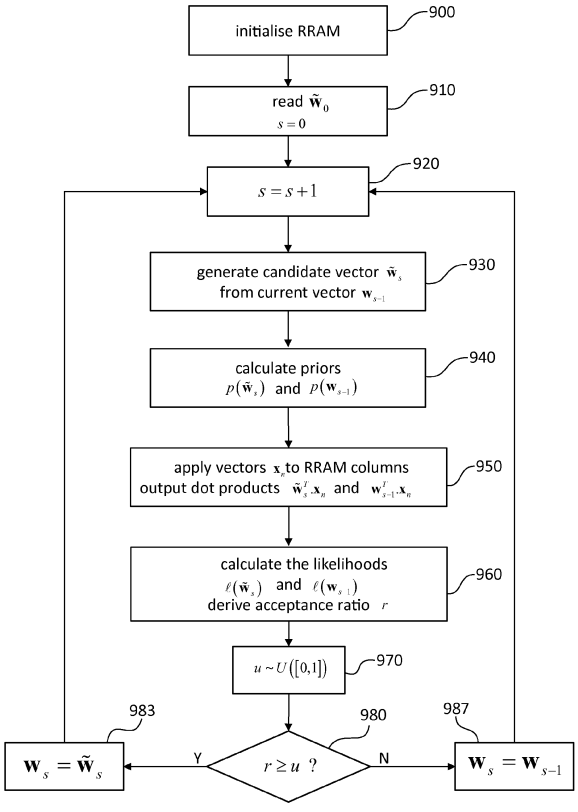

Fig. 9

EP 3 825 927 A1

Description

FIELD OF THE INVENTION

[0001] The present invention concerns the field of machine learning and more specifically the use of Markov Chain Monte Carlo (MCMC) sampling for machine learning. The invention also relates to the field of resistive Random Access Memories (RRAMs).

BACKGROUND OF THE INVENTION

[0002] Classification is a common problem within machine learning. Basically, classifying amounts to predicting a qualitative response for an observation i.e. assigning the observation to a category or a class. Many classification techniques are available to predict a qualitative response. A very popular supervised classification method is logistic regression.

[0003] Consider an example of dataset as illustrated in Fig. 1. Each element of the dataset can be described by two attributes (or features) x_1,x_2 and has a binary response t={0,1}. In other words, each element can be classified as belonging to a first class, t = 0, (points plotted as a circle) or as belonging to a second class, t = 0 (points plotted as a square).

[0004] The data elements of the dataset can be represented by vectors $\mathbf{x}_n$, the size of which being the number M of attributes (here M = 2). These vectors can be stacked in a matrix $\mathbf{X}=\left(\mathbf{x}_1^T,\mathbf{x}_2^T,...,\mathbf{x}_N^T\right)^T$ where N is the number of elements in the dataset, thereby representing the whole dataset. Similarly, the responses of the elements of the dataset can be stacked in a vector $\mathbf{t}=(t_1,t_2,...,t_N)^T$.

[0005] We look for a linear model which would allow us to classify the data elements of the dataset according to their known responses and predict the class of a new observation $\mathbf{x}_{new}$. More specifically, we assume that for any data element $\mathbf{x}_n$ of the dataset the probability that it belongs to the first class (T_n = 0) decreases with the dot product $\mathbf{w}^T\mathbf{x}_n$ where $\mathbf{w}$ is a vector, hereinafter referred to as parameter vector capturing the parameters of the model. Conversely, we assume that the probability that data element $\mathbf{x}_n$ belongs to the second class (T_n = 1) increases with the dot product. A simple way to convert the dot product into a probability value is to squash the dot product into the range [0,1] with a sigmoid function (a.k.a. standard logistic function), according to the logit model:

$$P(T_n=1|\mathbf{x}_n)=\frac{1}{1+\exp\left(-\mathbf{w}^T\mathbf{x}_n\right)} \tag{1-1}$$

and therefore:

$$P(T_n=0|\mathbf{x}_n)=1-P(T_n=1|\mathbf{x}_n)=\frac{\exp\left(-\mathbf{w}^T\mathbf{x}_n\right)}{1+\exp\left(-\mathbf{w}^T\mathbf{x}_n\right)} \tag{1-2}$$

[0006] Following a Bayesian approach, we now consider the parameter vector as a random vector and are interested in finding the posterior probability density (also called posterior density or, simply, posterior) knowing the dataset elements and their respective classes, that is $p(\mathbf{w}|\mathbf{t},\mathbf{X})$. The posterior can be obtained from the prior probability density, $p(\mathbf{w})$, (also called prior density or, simply, prior), the likelihood $p(\mathbf{t}|\mathbf{X},\mathbf{w})$ and the marginal likelihood, $p(\mathbf{t}|\mathbf{X})$, according to Bayes' rule:

$$p\left(\mathbf{w}|\mathbf{t},\mathbf{X}\right)=\frac{p\left(\mathbf{t}|\mathbf{X},\mathbf{w}\right)p(\mathbf{w})}{p\left(\mathbf{t}|\mathbf{X}\right)} \tag{2}$$

where the marginal likelihood is obtained by $p(\mathbf{|X})=\int p(\mathbf{t}|\mathbf{X},\mathbf{w})p(\mathbf{w})d\mathbf{w}$.

[0007] In the following, the posterior $p(\mathbf{w}|\mathbf{t},\mathbf{X})$ will simply be denoted $\pi(\mathbf{w})$ and the likelihood of parameter vector $\mathbf{w}$, $p(\mathbf{t}|\mathbf{X},\mathbf{w})$, will be denoted $\ell(\mathbf{w})$. Since $p(\mathbf{t}|\mathbf{X})$ is a constant depending only upon the dataset, the posterior is proportional to the product of the likelihood with the prior:

$$\pi(\mathbf{w}) = Z.\ell(\mathbf{w}).p(\mathbf{w}) \tag{3}$$

where Z is a constant.

[0008] It should be noted that the terms posterior and prior always refer to the available observations (the dataset **X**) and their respective responses (**t**).

[0009] Assuming that the elements of **t** are conditionally independent, the likelihood ℓ**(w)** can be expressed as follows:

$$\ell(\mathbf{w}) = p\left(\mathbf{t}\middle|\mathbf{X},\mathbf{w}\right) = \prod_{n=1}^{N} p\left(t_n\middle|x_n,\mathbf{w}\right) = \prod_{n=1}^{N}\left(p\left(T_n=1\middle|x_n,\mathbf{w}\right)\right)^{t_n}\left(p\left(T_n=0\middle|x_n,\mathbf{w}\right)\right)^{1-t_n} \tag{4}$$

and therefore, substituting expressions (1-1) and (1-2) into (4):

$$\ell(\mathbf{w}) = \prod_{n=1}^{N}\left(\frac{1}{1+\exp\left(-\mathbf{w}^T\mathbf{x}_n\right)}\right)^{t_n}\left(\frac{\exp\left(-\mathbf{w}^T\mathbf{x}_n\right)}{1+\exp\left(-\mathbf{w}^T\mathbf{x}_n\right)}\right)^{1-t_n} \tag{5}$$

[0010] Even assuming a known distribution *e.g.* a Gaussian distribution for the prior π**(w)**, it is generally impossible to compute the denominator Z^{-1} of (2) because the integration of the likelihood p**(t|X,w)** for obtaining the marginal likelihood p**(t|X)** is not analytically tractable.

[0011] Several options are available at this stage. According to a first option, one may look for the parameter vector, $\hat{\mathbf{w}}_{MAP}$ (where MAP stands for *Maximum A Posteriori*) achieving the maximum value of the posterior, π**(w)** and predict the class of a new observation according to (1-1) and (1-2) with the MAP parameter vector, $\hat{\mathbf{w}}_{MAP}$. According to a second option, referred to as Laplace approximation, one may try to approximate the posterior around its maximum value by a gaussian distribution. A third option, which better captures the distribution of the parameter vector is to sample the posterior π**(w)** knowing the likelihood ℓ**(w)** and the prior p(w) and availing of the proportionality relationship (3). Sampling means drawing samples of the parameter vector **w** according to the posterior π**(w)**. Once *Ns* samples of the posterior have been obtained, $\mathbf{w}_s$, s =1,...,*Ns*, namely once we have got *Ns* models we can predict the class of a new observation $\mathbf{x}_{new}$ (unseen data point) by calculating the average:

$$P(T_n=1\middle|\mathbf{x}_{new}) = \frac{1}{Ns}\sum_{s=1}^{Ns}\frac{1}{1+\exp\left(-\mathbf{w}_s^T\mathbf{x}_{new}\right)} \tag{6-1}$$

and therefore:

$$P(T_n=0\middle|\mathbf{x}_{new}) = 1-P(T_n=1\middle|\mathbf{x}_{new}) = \frac{1}{Ns}\sum_{s=1}^{Ns}\frac{\exp\left(-\mathbf{w}_s^T\mathbf{x}_{new}\right)}{1+\exp\left(-\mathbf{w}_s^T\mathbf{x}_{new}\right)} \tag{6-2}$$

[0012] Each instance of $\mathbf{w}_s$ capturing the model parameters is associated with a linear decision boundary in the space of attributes as shown in Fig. 2. More generally, each sample $\mathbf{w}_s$ defines a decision boundary is the form of a hyperplane in the space of attributes.

[0013] It follows from expression (6-1) and (6-2) that we can train a logistic regression classifier on the data set **X** with known labels **t** provided we are able to sample the posterior π**(w)**.

[0014] The sampling of π**(w)** can be achieved by a so-called MCMC (Markov Chain Monte Carlo) technique allowing to explore the parameter space according to a Markov stochastic process. A popular sampling algorithm using this technique is the Metropolis-Hastings algorithm which is outlined further below. A detailed presentation of the Metropolis-Hastings algorithm and its application to machine learning can be found for instance in the book by Simon Rogers and Mark Girolami entitled "First course in Machine Learning", Chapter IV, second edition, Chapman & Hall/ CRC Press, 2017, or in the article of C. Andrieu et al. entitled "An introduction to MCMC for machine learning" published in Machine

Learning, vol. 5, pages 5-43, 2003.

[0015] A flowchart of the Metropolis-Hastings algorithm is illustrated in Fig. 3.

[0016] It assumes that a conditional probability density for moving from a parameter vector to a next one is defined beforehand.

[0017] The algorithm starts at step 310 with an arbitrary parameter vector $\mathbf{w}_0$, and by initialising an iteration counter $s = 0$.

[0018] At step 320, the iteration counter is incremented by one, that is $s=s+1$.

[0019] At step 330, a new candidate parameter vector, $\tilde{\mathbf{w}}_s$, is generated by using the conditional probability density $p(\tilde{\mathbf{w}}_s|\mathbf{w}_{s-1})$ mentioned above. Without prejudice to generality, this conditional probability density can be chosen Gaussian, *i.e.* $p(\tilde{\mathbf{w}}_s|\mathbf{w}_{s-1})\sim N(\mathbf{w}_{s-1},\Sigma)$ where Σ is a predetermined covariance matrix. Σ is chosen diagonal and its eigenvalues are chosen commensurate with the parameter space to explore. It should be noted that the conditional probability density is simply here to define a random walk through the parameter space and is not related to the posterior $\pi(\mathbf{w})$.

[0020] Fig. 4 illustrates an example of random walk in the model space performed by the Metropolis-Hastings algorithm. The average size of a move from one sample to the next is given by the eigenvalues of the covariance matrix.

[0021] Returning to the flowchart of Fig. 3, step 340 calculates an acceptance ratio, *r*, defined as the ratio of the posterior density at the candidate sample $\tilde{\mathbf{w}}_s$ to that at the previous sample $\mathbf{w}_{s-1}$, divided by the ratio of the proposed densities, namely:

$$r = \frac{\pi(\tilde{\mathbf{w}}_s)}{\pi(\mathbf{w}_{s-1})} \cdot \frac{p(\mathbf{w}_{s-1}|\tilde{\mathbf{w}}_s)}{p(\tilde{\mathbf{w}}_s|\mathbf{w}_{s-1})} = \frac{\pi(\tilde{\mathbf{w}}_s)}{\pi(\mathbf{w}_{s-1})} \tag{7}$$

since the Gaussian distribution is symmetrical. The acceptance ratio *r* appraises the suitability of the candidate vector $\tilde{\mathbf{w}}_s$ to better classify the data of the dataset than vector $\mathbf{w}_{s-1}$ output by the previous iteration.

[0022] Although the posteriors cannot be calculated due to the presence of the constant *Z*, their ratio can be easily obtained from expression (3) as the product of the ratio of the priors with the ratio of likelihoods:

$$r = \frac{\pi(\tilde{\mathbf{w}}_s)}{\pi(\mathbf{w}_{s-1})} = \frac{\ell(\tilde{\mathbf{w}}_s)}{\ell(\mathbf{w}_{s-1})} \cdot \frac{p(\tilde{\mathbf{w}}_s)}{p(\mathbf{w}_{s-1})} \tag{8}$$

in which we will assume that the priors are all Gaussian i.e. $p(\tilde{\mathbf{w}}_s)\sim N(0,\sigma^2\mathbf{I}_M)$ and $p(\mathbf{w}_{s-1}) \sim N(0,\sigma^2\mathbf{I}_M)$, where $\mathbf{I}_M$ is the identity matrix of size $M\times M$, *M* being the size of the data elements, that is the number of attributes.

[0023] It is checked at step 350 whether the acceptance ratio is greater than or equal to 1. In the affirmative, the algorithm jumps to step 353 where the candidate vector is accepted that is $\mathbf{w}_s = \tilde{\mathbf{w}}_s$. In the negative, the algorithm proceeds further with step 360 where a sample value, *u,* of a random variable *U* having a uniform probability density on the interval [0,1] is drawn.

[0024] It is then checked at step 370 whether the sample value, *u*, is lower than or equal than the acceptance ratio, *r.* In the affirmative, the algorithm jumps to step 353 where the candidate vector is accepted, that is $\mathbf{w}_s = \tilde{\mathbf{w}}_s$. However, in the negative, the candidate vector is rejected at step 357 and the parameter vector output by the iteration *s* is chosen equal to the one output by previous iteration, that is $\mathbf{w}_s = \mathbf{w}_{s-1}$.

[0025] In other words, if posterior density at $\tilde{\mathbf{w}}_s$ is higher that the posterior density at $\tilde{\mathbf{w}}_{s-1}$, the candidate vector is systematically retained whereas it is only conditionally retained otherwise. Hence, it should be understood that due to the rejection mechanism, several consecutive samples of the sequence $\mathbf{w}_s$, s = 1,..., *Ns* may be identical.

[0026] After having being updated by the candidate vector (step 353) or not (step 357), the algorithm goes back to step 320 for a new iteration.

[0027] Since, $u \leq 1$, it is important to note that step 350 can be skipped. It suffices then to compare the acceptance ratio, *r*, with *u* in 370, the parameter vector being updated with $\tilde{\mathbf{w}}_s$ if $u \leq r$ and being kept equal identical to the previous parameter vector $\mathbf{w}_{s-1}$ otherwise.

[0028] The algorithm stops when a predetermined stopping criterion (not represented) is met, e.g. when a predetermined number N_s of samples have been obtained.

[0029] The Metropolis-Hastings algorithm can be run on a traditional computer for training a logistic regression classifier. However, when the dataset is large or when the size (N_s) of the model is large, the algorithm requires a large size memory and powerful computer resources.

[0030] An object of the present invention is to propose a method for training a logistic regression classifier which can be carried out very efficiently on a hardware accelerator having very simple architecture.

BRIEF DESCRIPTION OF THE INVENTION

[0031] The present invention is defined by the appended independent claims. Various preferred embodiments are defined in the dependent claims.

BRIEF DESCRIPTION OF THE DRAWINGS

[0032] The present invention will be better understood from the description of the following embodiments, by way of illustration and in no way limitative thereto:

Fig. 1 schematically illustrates an example of a dataset defined by two features following a binary response model;
Fig. 2 schematically illustrates decision boundaries corresponding to instances of the model parameters for separating the two classes of response in Fig. 1;
Fig. 3 schematically illustrates the flow chart of the Metropolis-Hastings algorithm for sampling the model parameters;
Fig. 4 shows an example of random walk in the model space performed by the Metropolis-Hastings algorithm;
Fig. 5 shows examples of a probability density function of the resistance value of a cell in a low resistance state as well as examples of a probability density function of the resistance value of a cell in a high resistance state;
Fig. 6 shows the relationship between the SET programming current and the median low resistance value of a cell in an LRS state;
Fig. 7 shows the relationship between the RESET programming voltage and the median high resistance value of a cell in an HRS state;
Fig. 8 shows the structure of a RRAM which can be used for a training method according to a first embodiment of the invention;
Fig. 9 schematically illustrates the flowchart of a method for training a logistic regression classifier on a dataset by using the RRAM of Fig. 8, according to the first embodiment of the invention;
Fig. 10 shows a method for classifying new data with a RRAM based logistic regression classifier after it has been trained by a training method according to the first embodiment of the invention.
Fig. 11 shows the structure of a RRAM which can be used for a training method according to a second embodiment of the invention.

DETAILED DISCLOSURE OF PARTICULAR EMBODIMENTS

[0033] The idea at the basis of the present invention is to use a resistive switching RAM also simply called resistive memory (RRAM) for implementing the training of a logistic regression classifier according to the Metropolis-Hastings algorithm. More specifically, the invention makes use of the cycle to cycle (C2C) variability of the low-resistance after a SET operation or the high-resistance after a RESET operation in order to generate successive parameter vectors of the model.

[0034] We recall that a resistive switching RAM consists of non-volatile random access memory cells, each cell comprising a resistor made of dielectric which can be programmed either in a low resistance state (LRS) with a so-called SET operation or in a high resistance state (HRS) with a so-called RESET operation. During a SET operation, a strong electric field is applied to the cell, while limiting the current to a programming current value. This operation forms a conductive filament through the dielectric and brings the resistor to a low resistance value, R_{LO} which depends upon the programming current value.

[0035] Conversely, during a RESET operation, a programming voltage is applied to the cell with the same or opposite polarity than the one used for electroforming. This voltage breaks the filament and the resistor returns therefore to a high resistance value, R_{HI} which depends upon the programming voltage value.

[0036] A detailed description of RRAM can be found in the article by R. Carboni and D. Ielmini, entitled "Stochastic memory devices for security and computing", published in Adv. Electron. Mat., 2019, 5, 1900198, pages 1-27.

[0037] Once a cell has been programmed by a SET or a RESET operation, the resistance value (R_{LO} or R_{HI}) is stable in time until the next operation is performed. However, the low resistance value varies from one SET operation to the next, even if the programming current is kept constant. More specifically, for a given programming current, the low resistance R_{LO} can be considered as a random variable which exhibits a normal distribution over SET operations (cycle to cycle variability). Expressed otherwise, each time a SET operation is applied to the memory cell, the obtained resistance value is sampled from this normal distribution.

[0038] Similarly, the high resistance value varies from one RESET operation to the next, even if the programming voltage is kept constant. More specifically, for a given programming voltage, high resistance R_{HI} can be considered as a random variable which follows a log-normal distribution over the RESET operations (cycle to cycle variability). In other words, each time a RESET operation is applied to the memory cell, the obtained resistance value is sampled from this

log-normal distribution.

[0039] Fig. 5 shows, on its left side, examples of probability density function of R_{LO} for a cell in a low resistance state. The three examples shown correspond to three different RRAM cells. It also shows, on its right side, examples of probability density function of R_{HI} for a cell in a high resistance state HRS. The three examples shown correspond to the same three RRAM cells. The probability density function has been obtained by cycling 100 times each of the cells. The x-axis being represented in logarithm scale and the shape of R_{HI} distribution being Gaussian, this confirms that R_{HI} follows a log-normal distribution.

[0040] The median value of the R_{LO} distribution (or, equivalently, the mean value since the distribution is normal) depends on the programming current value during the SET operation as illustrated in Fig. 6. The x-axis indicates the programming current (in μA) according to a logarithmic scale whereas the y-axis indicates the median value R_{LO}^{median} (in Ω) according to a logarithmic scale. This dependency being linear in logscale, it follows that the median value of the low resistance follows a power law of the type $R_{LO}^{median} = a\left(I_{SET}\right)^{-\gamma}$ where *a* and γ are positive coefficients.

[0041] Furthermore, the median value of the R_{HI} distribution depends upon the programming voltage value during the RESET operation, as illustrated in Fig. 7. The x-axis indicates the programming voltage (in V) according to a linear scale whereas the y-axis indicates the median value, R_{HI}^{median} (in Ω) according to a logarithmic scale. This dependency being linear in semi-log scale, it follows that the median value of the high resistance follows an exponential law of the type $R_{HI}^{median} = b\exp\left(\lambda V_{RESET}\right)$ where *b* and λ are positive coefficients.

[0042] According to a first embodiment of the invention, the method for training a logistic regression classifier uses a resistive RAM structure as shown in Fig. 8.

[0043] The resistive RAM is comprised of word lines which are addressed by word select signals *row*[0],*row*[1],...,*row*[*P*]. Each word line is comprised of *M* cells where *M* stands for the number of attributes, each word line being intended to store a sample a parameter word of the model. Furthermore, each word line is associated with a counter of occurrences, the function of which will be described further below.

[0044] The cells of a column, *m* =1,...,*M*, can be fed with a data control signal, *col*[*m*]. Each cell comprises a FET transistor, the gate of which is controlled by a word select signal and the source of which is connected to a data control signal through a resistor, $R_{p,m}$. The drains of the FETs of a word line, *p* = 1,...,*P*, are all connected to the same output line, *out* [*p*].

[0045] During a write operation into a cell located at the intersection of row p and column m, word select signal *row*[*p*] is applied and a programming voltage is applied to *col*[*m*]. The programming can occur during a SET operation or a RESET operation as mentioned above.

[0046] During a read operation of a cell located at the intersection of row p and column m, a word select signal is applied at *row*[*p*] and a data control signal is applied at *col*[*m*]. Assuming that a voltage x_m is applied to *col*[*m*], the output line *output*[*p*] will output a current x_m / $R_{p,m}$ = $g_{p,m}x_m$, where $g_{p,m}$ is the conductance of the resistive RAM cell. If the RAM cell was programmed to store a value (of conductance) $w_{p,m}$, the output current would be $w_{p,m}{\cdot}x_m$. Hence, if all the cells of a line are read simultaneously, the output current of line *output*[*p*] will be the value of the dot product $\mathbf{w}^{\mathrm{T}}{}_p\mathbf{x}$, where $x = (x_1,...,x_M)^T$ and $\mathbf{w}_p$ is the parameter vector stored at line *p* .

[0047] The resistive RAM of Fig. 8 can be used as hardware accelerator for training a logistic regression classifier on a dataset as explained in the flowchart of Fig. 9.

[0048] We adopt here the same notations as in the introductory part of the application. Each element of the training dataset is represented by *M* attributes which can be stacked in a vector x of size *M* . Each element of the training dataset is labelled with a binary target *t* indicating whether the element in question belongs to a first class or to a second class.

[0049] The probability of an element **x** belonging to a class is modelled by a logistic function applied to a score $\mathbf{w}^T\mathbf{x}$ where **w** is a parameter vector of size *M*. More specifically, the probability of element **x** belonging to the first class is given by:

$$P(T_n = 1|\mathbf{x}) = \frac{1}{Ns}\sum_{s=1}^{Ns}\frac{1}{1+\exp\left(-\mathbf{w}_s^T\mathbf{x}\right)} \qquad (9\text{-}1)$$

whereas the probability of this element belonging to the second class is given by:

$$P(T_n = 0|\mathbf{x}) = \frac{1}{Ns}\sum_{s=1}^{Ns}\frac{\exp\left(-\mathbf{w}_s^T\mathbf{x}\right)}{1+\exp\left(-\mathbf{w}_s^T\mathbf{x}\right)} \quad (9\text{-}2)$$

where the parameter vectors $\mathbf{w}_s$, s= 1,...,Ns are obtained by MCMC sampling the posterior π**(w)**, that is knowing the elements of the dataset **X** and their respective labels.

[0050] At step 900, the RRAM is initialised by programming all the cells in either the first resistance state (LRS), according to a first variant or the second resistance state (HRS), according to a second variant that is by performing a SET operation (first variant) or a RESET operation (second variant) on all the cells of the RRAM. Preferably, the cells are programmed in the first resistance state for reasons that will become more apparent below.

[0051] At step 910, the first row of the RAM is read, that is the data $\tilde{w}_{0,1},...,\tilde{w}_{0,M}$ forming a first initial parameter vector $\tilde{\mathbf{w}}_0$. The counter of iterations s is initialized, s= 0. The counters of occurrences associated with the different rows are reset to zero.

[0052] The training method then enters an iterative loop, where s stands for the current iteration, $\mathbf{w}_{s-1}$ is the current parameter vector which has already been stored in the RRAM at the previous iteration and $\tilde{\mathbf{w}}_s$ is a candidate parameter vector at iteration s .

[0053] At step 920, the counter of iterations is incremented, s=s+1.

[0054] At step 930, a candidate parameter vector, $\tilde{\mathbf{w}}_s$, is generated from the current parameter vector, $\mathbf{w}_{s-1}$, by reading the cells of the current row and programming the cells of the next row with a SET operation (1st variant) or with a RESET operation (2nd variant). More specifically a current or voltage is read from the cell of current row j and applied as a voltage to the cell of the next row of the same column. By so doing, a candidate vector, $\tilde{\mathbf{w}}_s$, is generated with conditional probability $p(\tilde{\mathbf{w}}_s|\mathbf{w}_{s-1})$ according to a normal distribution law (1st variant), and stored temporarily at row j+1. As to the second variant, the current or voltage read from the cell current row j is generated with a conditional probability which follows a log-normal law. Preferably, the read current or voltage is applied to an exponential amplifying circuit (for example an operational amplifier having an exponential characteristic element, such as a diode or biased FET on its non-inverting input) before being applied to the cell of the next row and same column. The exponential amplifying circuit projects the log-normal distribution into a normal distribution.

[0055] It follows from this step that the resistances of the cells of the next row are derived from the resistances of the current row according to a Markovian process.

[0056] At step 940, the current parameter vector, $\mathbf{w}_{s-1}$, and the candidate parameter, $\tilde{\mathbf{w}}_s$, respectively stored at the current row and at the next row are read from the RRAM. The priors $p(\tilde{\mathbf{w}}_s)$ and $p(\mathbf{w}_{s-1})$ are calculated by a processor (not shown) from a predetermined gaussian law which may have been obtained during a calibration phase. Preferably, these values are calculated under a logarithm form, i.e. $\log(p(\tilde{\mathbf{w}}_s))$ and $\log(p(\mathbf{w}_{s-1}))$.

[0057] At step 950, the vectors $\mathbf{x}_n$, n=1,..,N of the dataset **X** are applied in turn to the set of columns of the RRAM. The output lines *output*[j] and *output*[j+1] respectively output the dot products $\tilde{\mathbf{W}}_s^T.\mathbf{x}_n$ and $\mathbf{W}_{s-1}^T.\mathbf{x}_n$ n=1,..,N, which are supplied to the processor.

[0058] At step 960, the processor computes the likelihoods $\ell(\tilde{\mathbf{w}}_s)$ and $\ell(\mathbf{w}_{s-1})$ from expression (5):

$$\ell\left(\tilde{\mathbf{w}}_s\right) = \prod_{n=1}^{N}\left(\frac{1}{1+\exp\left(-\tilde{\mathbf{w}}_s\mathbf{x}_n\right)}\right)^{t_n}\left(\frac{\exp\left(-\tilde{\mathbf{w}}_s{}^T\mathbf{x}_n\right)}{1+\exp\left(-\tilde{\mathbf{w}}_s{}^T\mathbf{x}_n\right)}\right)^{1-t_n} \quad (10\text{-}1)$$

$$\ell\left(\mathbf{w}_{s-1}\right) = \prod_{n=1}^{N}\left(\frac{1}{1+\exp\left(-\mathbf{w}_{s-1}\mathbf{x}_n\right)}\right)^{t_n}\left(\frac{\exp\left(-\mathbf{w}_{s-1}{}^T\mathbf{x}_n\right)}{1+\exp\left(-\mathbf{w}_{s-1}{}^T\mathbf{x}_n\right)}\right)^{1-t_n} \quad (10\text{-}2)$$

[0059] It is noted that for each term of product (10-1) or (10-2) only one of the two ratios has to be computed (namely the one corresponding to t_n =1).

[0060] Preferably, the log-likelihoods $\log(\ell(\tilde{\mathbf{w}}_s))$ and $\log(\ell(\mathbf{w}_{s-1}))$ are calculated instead as a sum of terms, only those being weighted by t_n = 1 needing to be computed.

[0061] The processor then computes the product of the ratio of the likelihoods $\frac{\ell(\tilde{\mathbf{w}}_s)}{\ell(\mathbf{w}_{s-1})}$ with the ratio of the priors $\frac{p(\tilde{\mathbf{w}}_s)}{p(\mathbf{w}_{s-1})}$ to obtain the acceptance ratio according to expression (8).

[0062] Preferably, the acceptance ratio can be computed under its logarithm form, log(*r*) as the sum of difference of log-likelihoods and difference of log of priors.

[0063] At step 970, a random value *u* is sampled from a random variable *U* exhibiting a uniform probability density law over range [0,1]. The random value can be generated by the processor. Alternatively, it can be generated by a dedicated row of the RRAM, for example *row*[*P*] as shown in Fig. 8. In order to obtain a uniform distribution, a voltage close to the threshold SET voltage is applied as data control signal to the columns *col*[*m*] *m*=1,...,*M*. In such instance, each cell has a probability 1/2 to be in the LRS state and a probability 1/2 to be in the HRS state. Upon reading the word stored at *row*[*P*], the output line *output*[*P*] supplies current is supplied which is uniformly generated between 0 and a predetermined value I_{max}, from which the random value *u* of uniform law over range [0,1] can be derived.

[0064] At step 980, the acceptance ratio *r* is compared with the random value *u.* In practice, the comparison can be performed between the logarithmic values log(*r*) and log(*u*).

[0065] If $r \geq u$, the candidate parameter vector $\tilde{\mathbf{w}}_s$ is retained as the new current vector at step 983, that is $\mathbf{w}_s = \tilde{\mathbf{w}}_s$ and the new current row is *j*+1. The counter of occurrences associated with row *j*+1 is set to 1.

[0066] Else, if $r < u$, the candidate parameter is rejected at step 987, that is $\mathbf{w}_s = \mathbf{w}_{s-1}$, the current row is kept at *j* and the counter of occurrences associated with row *j* is incremented by one.

[0067] In both cases, the algorithm checks whether a stopping criterion is met, for example whether $s = N_s$, that is whether a maximum number of samples has been reached or if the RRAM is full. In the negative, the algorithm goes back to step 920. Otherwise, it stops. According to a variant, instead of stopping when the RRAM is full, that is when the last row has been reached, the algorithm loops back to the first row and goes on as before. This variant is of interest when convergence to the posterior is slow.

[0068] Once the logistic regression classifier has been trained on the training dataset, that is, in practice, when the N_s samples of parameter vector have been stored in the RRAM, the RRAM can be used for the classification of new observation as shown in Fig. 10.

[0069] Fig. 10 shows a method for classifying new observation with a RRAM based logistic regression classifier after it has been trained by a training method according to the first embodiment of the invention.

[0070] The RRAM contains *Ns* samples of the model parameter vector $\mathbf{w}_s$, $s = 1,...,Ns$. It is nevertheless important to understand that the same sample value may occur several times, the occurrences of the same sample value being stored in the same row. More specifically, if we denote v_j the integer stored in the counter of occurrences associated with row *j* and *J* the number of rows where a sample has been stored (i.e. the rows for which the associated counters are not equal to zero), we have the relationship:

$$N_s = \sum_{j=1}^{J} v_j \tag{11}$$

[0071] At step 1000, when a new observation $\mathbf{x}_{new}$ is to be classified, its components are applied as data control signals to columns 1,...,*M*. These control signals are in practice voltages proportional to the components of $\mathbf{x}_{new}$.

[0072] The classifying then enters an iterative loop, the loop being iterated as long as an occurrence counter is not zero.

[0073] At step 1010, the first *J* rows are selected and output lines *output*[1],*output*[2],...,*output*[*J*] are read in parallel or in series by the processor. These output lines provide the dot products $\mathbf{w}_j^T \mathbf{x}_{new}$, $j = 1,..., J$ where $\mathbf{w}_j$ is stored at row *j*. According to a variant, the first *B* rows of the RRAM can be ignored or discarded, these first *B* rows corresponding to a burn-in period of the training phase, corresponding to convergence to the posterior. In such instance, the *J* rows would be following these *B* rows.

[0074] After the output lines have been read, all the counters of occurrences that are not equal to zero are decremented by one at step 1020.

[0075] It is the checked at 1030 whether $\exists j \in \{1,...,J\}$ such as $v_j > 0$. In the affirmative the algorithm goes back to step 1010.

[0076] Conversely, in the negative, the iterative loop is exited. The reading of the output lines provides the sequence of samples (of the parameter vector) $\mathbf{w}_s$, $s = 1,...,Ns$.

[0077] Alternatively, instead of sequentially performing reading iterations until all the counters are equal to zero, the output of each line *output*[*j*] may be amplified by an amplifier whose gain is controlled by the value v_j stored in the corresponding counter.

[0078] The processor calculates at step 1040 an average logistic regression value over the samples, according to expression (6-1) or (6-2).

[0079] Finally, at step 1050, it is decided upon comparing $\boldsymbol{P}(\boldsymbol{T}_n = 1|\mathbf{x}_{new})$ or $\boldsymbol{P}(\boldsymbol{T}_n = 0|\mathbf{x}_{new})$ with a predetermined threshold value (e.g. 1/2) whether data element $\mathbf{x}_{new}$ belongs to the first class, 1051, or the second class, 1052.

[0080] Fig. 11 shows the structure of a resistive RAM which can be used for a training method according to a second embodiment of the invention.

[0081] The structure of this resistive RAM differs from the one used in the first embodiment by the fact that the parameter vectors of the models are stored as differential pairs. More precisely, each parameter vector $\mathbf{w}_j$ is represented by a pair $\left(\mathbf{w}_j^+, \mathbf{w}_j^-\right)$ with $\mathbf{w}_j = \mathbf{w}_j^+ - \mathbf{w}_j^-$ where the first part, $\mathbf{w}_j^+$, and the second part, $\mathbf{w}_j^-$, of parameter vector $\mathbf{w}_j$ are generated and stored separately in the RRAM.

[0082] More specifically, the resistive RAM is comprised of word lines which are addressed by word select signals *row*[0],*row*[1],...,*row*[*P*]. Each word select signal, *row*[*j*] controls in fact two sub-rows: a first sub-row storing the first part of the parameter vector, $\mathbf{w}_j^+$, and a second sub-row containing the second part of the parameter vector, $\mathbf{w}_j^-$. The outputs of the cells of the first sub-row of row *j* are connected to a first sub-row output line, *sub_output*$^+$[*j*] and the outputs of the cells of the second sub-row are connected to a second sub-row output line *sub_output*$^-$[*j*]. The first and second sub-row output lines are connected to the inputs of a subtractor *sub*[*j*].

[0083] The first sub-row of row *j* stores $\mathbf{w}_j^+$ and the second sub-row of row *j* stores $\mathbf{w}_j^-$. Hence, when the components of a vector **x** are applied to the data control lines and row *j* is selected, the first sub-row output line outputs $\left(\mathbf{w}_j^+\right)^T \mathbf{x}$ and the second sub-row output line outputs $\left(\mathbf{w}_j^-\right)^T \mathbf{x}$. The output of subtractor *sub*[*j*] is therefore $\left(\mathbf{w}_j^+\right)^T \mathbf{x} - \left(\mathbf{w}_j^-\right)^T \mathbf{x} = \mathbf{w}_j^T \mathbf{x}$.

[0084] The last row of the RRAM can be dedicated to an RNG as in the first embodiment. Alternatively, the RNG can be located off the RRAM.

[0085] The RRAM of Fig. 11 can be used for training a logistic regression classifier along the same lines as those described in relation with Fig. 9. Basically, for generating a parameter vector of a model, the first part, $\mathbf{w}_j^+$, and the second part, $\mathbf{w}_j^-$, the parameter vector are generated and respectively stored in the first sub-row and second sub-row of row *j*. However, a single counter of occurrences is associated to each row (and not each sub-row). Hence, when a new candidate vector, $\tilde{\mathbf{w}}_s$ is generated, it first part $\tilde{\mathbf{w}}_s^+$ and its second part $\tilde{\mathbf{w}}_s^-$ are respectively stored in the first sub-row and second sub-row of the next row, *j*+1. Otherwise, the counter of occurrences associated with the row *j* is incremented.

[0086] The first variant (programming with the SET operation) and the second variant (programming with the RESET operation) envisaged for the first embodiment equally apply to the second embodiment.

[0087] The present invention has been described in the context of the training of a logistic regression classifier, which classifies data into a class *C* (first class) or its complementary $\overline{C}$ (second class), the class the data belongs to being identified by a binary target value. The man skilled in the art will nevertheless understand that the invention can be extended to the training of a classifier which classifies data into a plurality of independent classes C_1, C_2,..., C_Q or the respective complementary classes. The RRAM is then divided in *Q* areas, each area being dedicated to a class C_q and being associated with a corresponding logistic regression classifier. Alternatively, *Q* RRAMs can be used, each RRAM being dedicated to a class C_q and being associated with a corresponding logistic regression classifier.

[0088] Finally, the training method according to the invention extends to a multinomial logistic classifier by using a softmax model instead of the logit model described above.

Claims

1. Method for training a logistic regression classifier on a dataset, each element of the dataset being represented by a vector **x** of *M* attributes, hereinafter referred to as components, and being labelled with a binary target indicating whether the element belongs to a first class or a second class, the probability of an element **x** belonging to a class being modelled by a logistic function applied to a score $\mathbf{w}^T\mathbf{x}$ of said element, where **w** is a parameter vector of size *M* representing the model, said method comprising the sampling of said parameter vector according to its posterior probability density with respect to the data set, and being **characterized by**:

   providing a resistive RAM, each row of the resistive RAM comprising *M* cells which can be programmed in a first resistance state or a second resistance state, each row being intended to store a parameter vector, a counter of occurrences being associated with each row;

      initializing said resistive RAM by setting all cells in said first resistance state;
      reading an initial vector $\tilde{\mathbf{w}}_0$ from the first row and entering an iterative loop comprising:

         (a) generating a candidate parameter vector, $\tilde{\mathbf{w}}_s$, from the current parameter vector, $\mathbf{w}_{s-1}$, by reading the cells of the current row and programming the resistances of the cells of the next row, the resistances of the cells of the next row being derived from the resistances of the cells of the current row according to a Markovian process;
         (b) computing an acceptance ratio, *r*, of the candidate parameter vector, $\tilde{\mathbf{w}}_s$, as the ratio of the posterior densities of $\tilde{\mathbf{w}}_s$ and $\mathbf{w}_{s-1}$, with respect to the dataset;
         (c) drawing a value *u* from a uniform distribution between 0 and 1 and comparing it with the acceptance ratio, *r*;

            (d1) if $u \leq r$ taking the next row as the current row and jumping to step (a);
            (d2) else, keeping the current row, incrementing the counter of occurrences associated therewith and jumping to step (a);

   the training method being stopped when a predetermined stopping criterion is met.

2. Method for training a logistic regression classifier according to claim 1, **characterised in that** the first resistance state is a high resistance state and the second resistance state is a low resistance state, and that, at step (a), the current or voltage read from a cell of the current row is used for applying a voltage to a corresponding cell of the next row while programming this cell in the low resistance state.

3. Method for training a logistic regression classifier according to claim 1, **characterised in that** the first resistance state is a low resistance state and the second resistance state is a high resistance state, and that, at step (a), the current or voltage read from a cell of the current row is amplified by an exponentiation circuit before being applied as a voltage to a corresponding cell of the next row while programming this cell in the high resistance state.

4. Method for training a logistic regression classifier according to claim 3, **characterised in that** the value *u* from said uniform distribution is obtained by applying a predetermined voltage to all the cells of a dedicated row, the outputs of the dedicated row being connected to an output line, said predetermined voltage being chosen close to the threshold voltage for programming the cells in a high resistance state, the value *u* being obtained from the output of said output line.

5. Method for training a logistic regression classifier according to any of the preceding claims, **characterised in that** the ratio of the posterior densities of $\tilde{\mathbf{w}}_s$ and $\mathbf{w}_{s-1}$ are obtained as the product of the ratio of the likelihoods $\frac{\ell(\tilde{\mathbf{w}}_s)}{\ell(\mathbf{w}_{s-1})}$ with respect to the dataset and the ratio of the priors $\frac{p(\tilde{\mathbf{w}}_s)}{p(\mathbf{w}_{s-1})}$.

6. Method for training a logistic regression classifier according to claim 5, **characterised in that** the likelihood of the candidate parameter vector $\ell(\tilde{\mathbf{w}}_s)$ is calculated as

$$\ell\left(\tilde{\mathbf{w}}_s\right)=\prod_{n=1}^{N}\left(\frac{1}{1+\exp\left(-\tilde{\mathbf{w}}_s\mathbf{x}_n\right)}\right)^{t_n}\left(\frac{\exp\left(-\tilde{\mathbf{w}}_s{}^T\mathbf{x}_n\right)}{1+\exp\left(-\tilde{\mathbf{w}}_s{}^T\mathbf{x}_n\right)}\right)^{1-t_n}$$

and the likelihood of the current parameter vector

$$\ell\left(\mathbf{w}_{s-1}\right)=\prod_{n=1}^{N}\left(\frac{1}{1+\exp\left(-\mathbf{w}_{s-1}\mathbf{x}_n\right)}\right)^{t_n}\left(\frac{\exp\left(-\mathbf{w}_{s-1}{}^T\mathbf{x}_n\right)}{1+\exp\left(-\mathbf{w}_{s-1}{}^T\mathbf{x}_n\right)}\right)^{1-t_n}$$

where $\mathbf{x}_n$, n=1,...,N are vectors representing data elements of the dataset, the dot product $\tilde{\mathbf{w}}_s{}^T\mathbf{x}_n$ being obtained by applying the components of vector $\mathbf{x}_n$ to the cells of the row containing candidate parameter vector $\tilde{\mathbf{w}}_s$, and by reading the common output line to which the cells of the row containing $\tilde{\mathbf{w}}_s$ are connected, the dot product $\mathbf{w}_{s-1}{}^T\mathbf{x}_n$ being obtained by applying the components of vector $\mathbf{x}_n$ to the cells of the row containing current parameter vector $\mathbf{w}_{s-1}$, and by reading the common output line to which the cells of the row containing $\mathbf{w}_{s-1}$ are connected.

7. Method for training a logistic regression classifier according to claims 1 to 6, **characterised in that** the predetermined stopping criterion is met when a predetermined number of samples (Ns) is reached or when the resistive RAM is full.

8. Method for classifying an observation with a logistic regression classifier trained according to claims 1 to 7, observation to be classified being represented by a vector $\mathbf{x}_{new}$ of M components, **characterised by**

respectively applying the M components to the M cells of each row of the resistive RAM storing a parameter vector;

(a') reading the output line associated with each row for which the counter of occurrences is not null, the reading of the output line providing the dot product $\mathbf{w}_s^T\mathbf{x}_{new}$ where $\mathbf{w}_s$ is the parameter vector stored at this row;
(b') decrementing the respective counters of occurrences of all the output lines that have been read at step (a');

steps (a') and (b') being repeated until all the counters of occurrences have reached zero, thereby obtaining a sequence of Ns parameter vectors, $\mathbf{w}_s^T\mathbf{x}_{new}$, s = 1,...,Ns.

computing the average $$P(T_n=1|\mathbf{x}_{new})=\frac{1}{Ns}\sum_{s=1}^{Ns}\frac{1}{1+\exp\left(-\mathbf{w}_s^T\mathbf{x}_{new}\right)}$$ and comparing it with a predetermined threshold value in order to determine whether said observation belongs to the first class of the second class.

9. Method for training a logistic regression classifier on a dataset, each element of the dataset being represented by a vector $\mathbf{x}$ of M attributes and being labelled with a binary target indicating whether the element belongs to a first class or a second class, the probability of an element $\mathbf{x}$ belonging to a class being modelled by a logistic function applied to a score $\mathbf{w}^T\mathbf{x}$ of said element, where $\mathbf{w}$ is a parameter vector of size M representing the model, said method comprising the sampling of said parameter vector according to its posterior probability density with respect to the dataset, said method being **characterized by**:

providing a resistive RAM, each row of the resistive RAM comprising a first sub-row and a second sub-row , each sub-row comprising M cells which can be programmed in a first resistance state or a second resistance state, the first sub-row being configured to store a first part of a parameter vector, the second sub-row being configured to store a second part of the same parameter vector, this parameter vector being obtained as the difference between said first part and said second part, and a counter of occurrences being associated with

each row;

initializing said resistive RAM by setting all cells in said first resistance state;
reading the first part and second part of an initial vector $\tilde{\mathbf{w}}_0$ from the first row and entering an iterative loop comprising:

(a) generating the first part and second part of a candidate parameter vector, $\tilde{\mathbf{w}}_s$, respectively from the first part and second part of current parameter vector, $\mathbf{w}_{s-1}$, by reading the cells of the first and second sub-row of current row and programming the resistances of the first and second sub-row of next row, the resistances of the cells of the first and second sub-row of next row being derived from the resistances of the cells of the first and second sub-row of current row according to a Markovian process;
(b) calculating the candidate parameter vector as the difference between its first part and second part; calculating the current parameter vector as the difference between its first part and second part; computing an acceptance ratio, r, of the candidate parameter vector, $\tilde{\mathbf{w}}_s$, as the ratio of the posterior densities of $\tilde{\mathbf{w}}_s$ and $\mathbf{w}_{s-1}$, with respect to the dataset ;
(c) drawing a value u from a uniform distribution between 0 and 1 and comparing it with the acceptance ratio, r;

(d1) if $u \leq r$ taking the next row as the current row and jumping to step (a);
(d2) else, keeping the current row, incrementing the counter of occurrences associated therewith and jumping to step (a);

the training method being stopped when a predetermined stopping criterion is met.

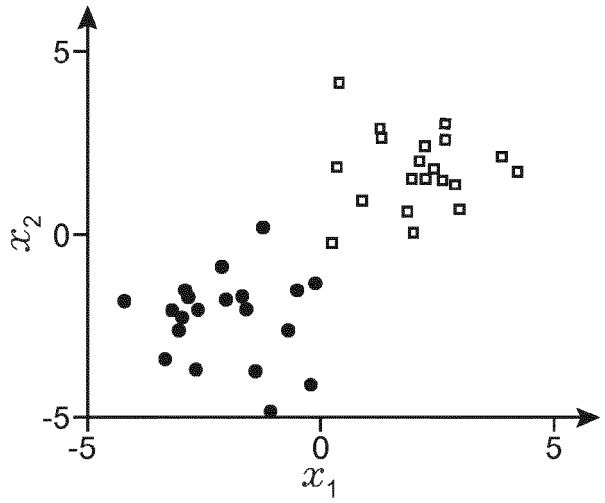

Fig. 1

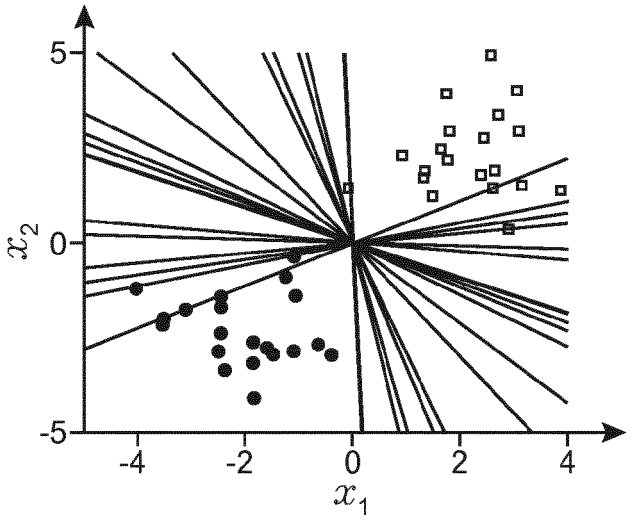

Fig. 2

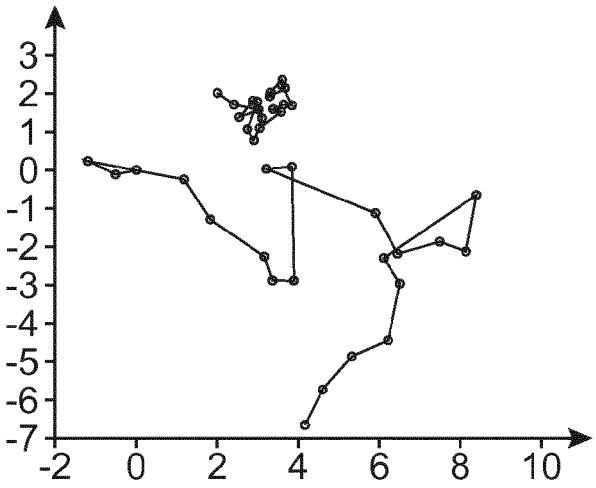

Fig. 4

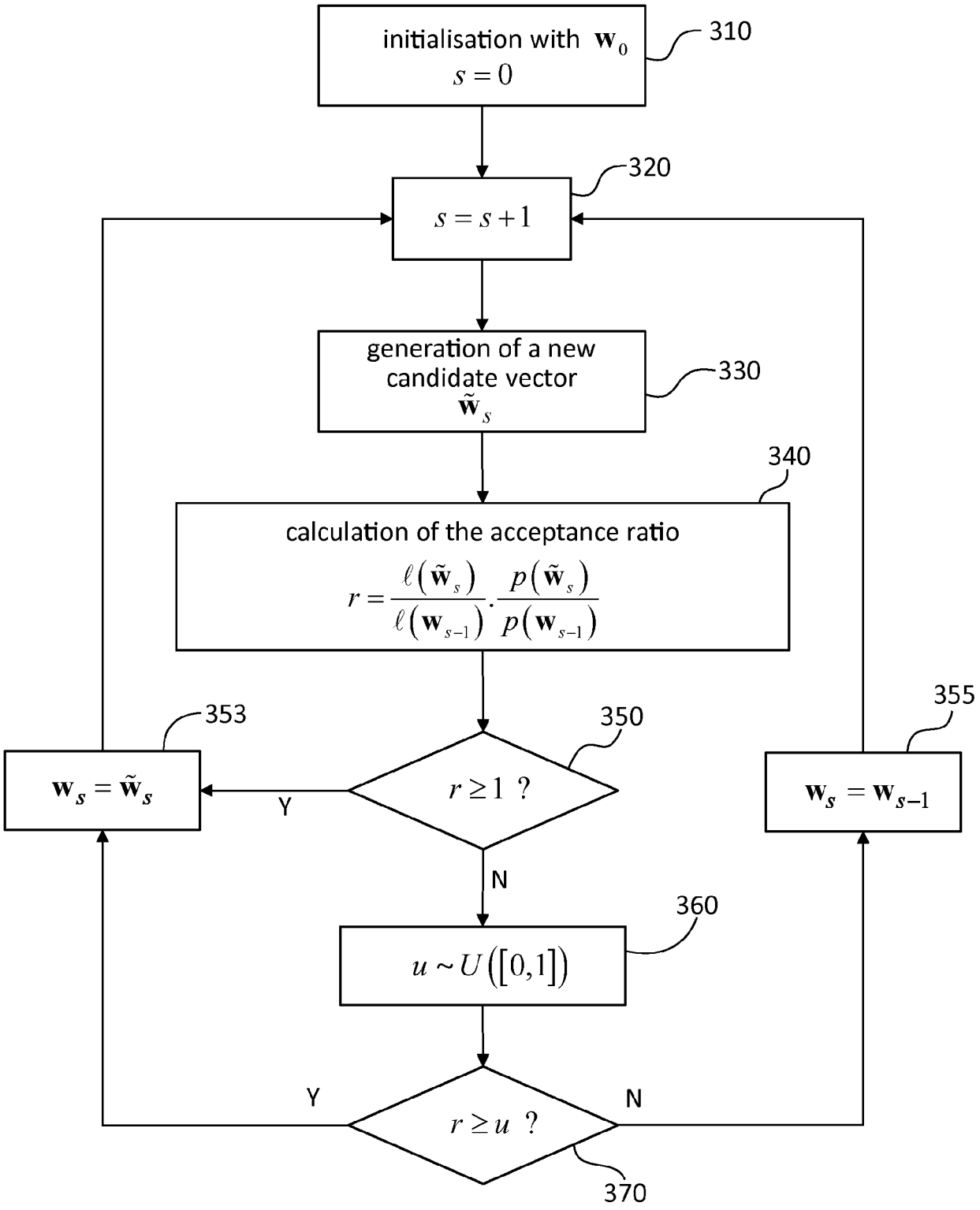
initialisation with $\mathbf{w}_0$
$s = 0$
310
320
$s = s + 1$
generation of a new candidate vector $\tilde{\mathbf{w}}_s$
330
340
calculation of the acceptance ratio
$r = \frac{\ell(\tilde{\mathbf{w}}_s)}{\ell(\mathbf{w}_{s-1})} \cdot \frac{p(\tilde{\mathbf{w}}_s)}{p(\mathbf{w}_{s-1})}$
350
$r \geq 1$?
Y
N
353
$\mathbf{w}_s = \tilde{\mathbf{w}}_s$
355
$\mathbf{w}_s = \mathbf{w}_{s-1}$
360
$u \sim U([0,1])$
Y
N
$r \geq u$?
370

Fig. 3

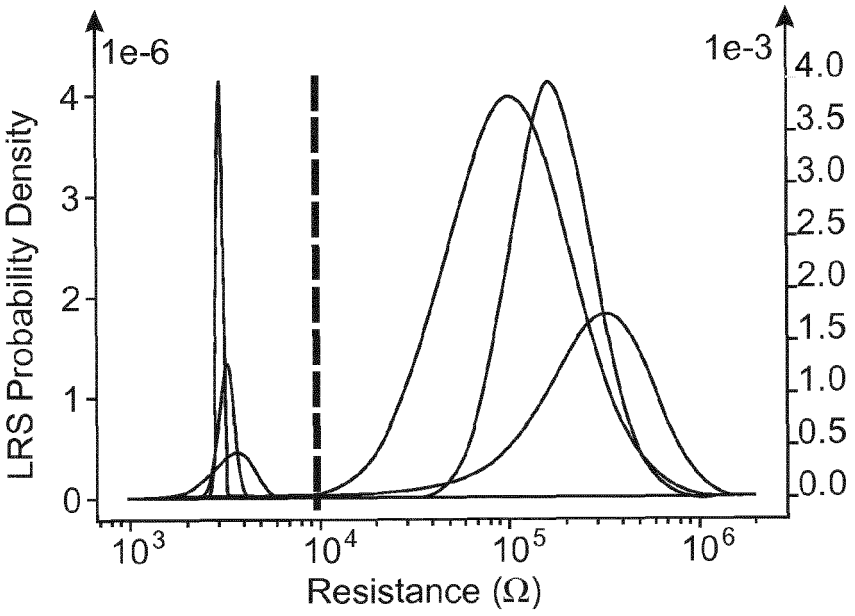

Fig. 5

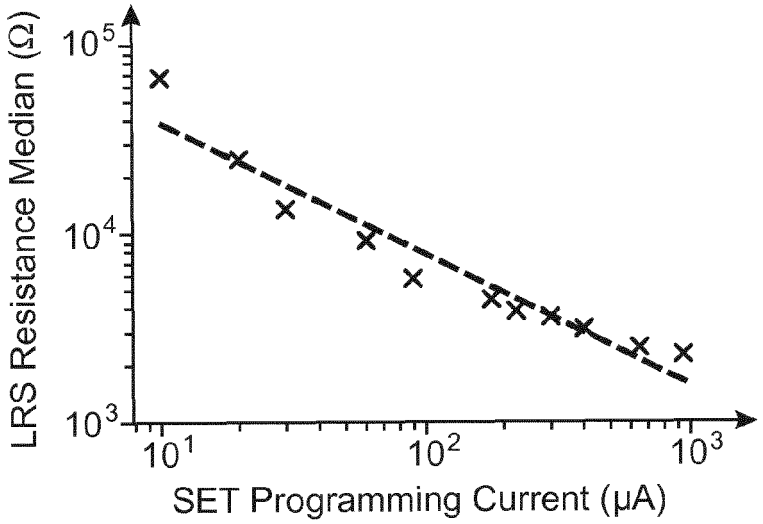

Fig. 6

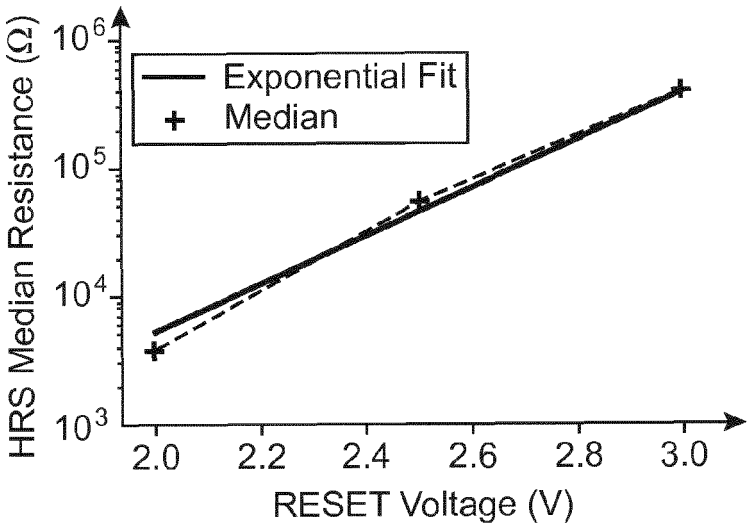

Fig. 7

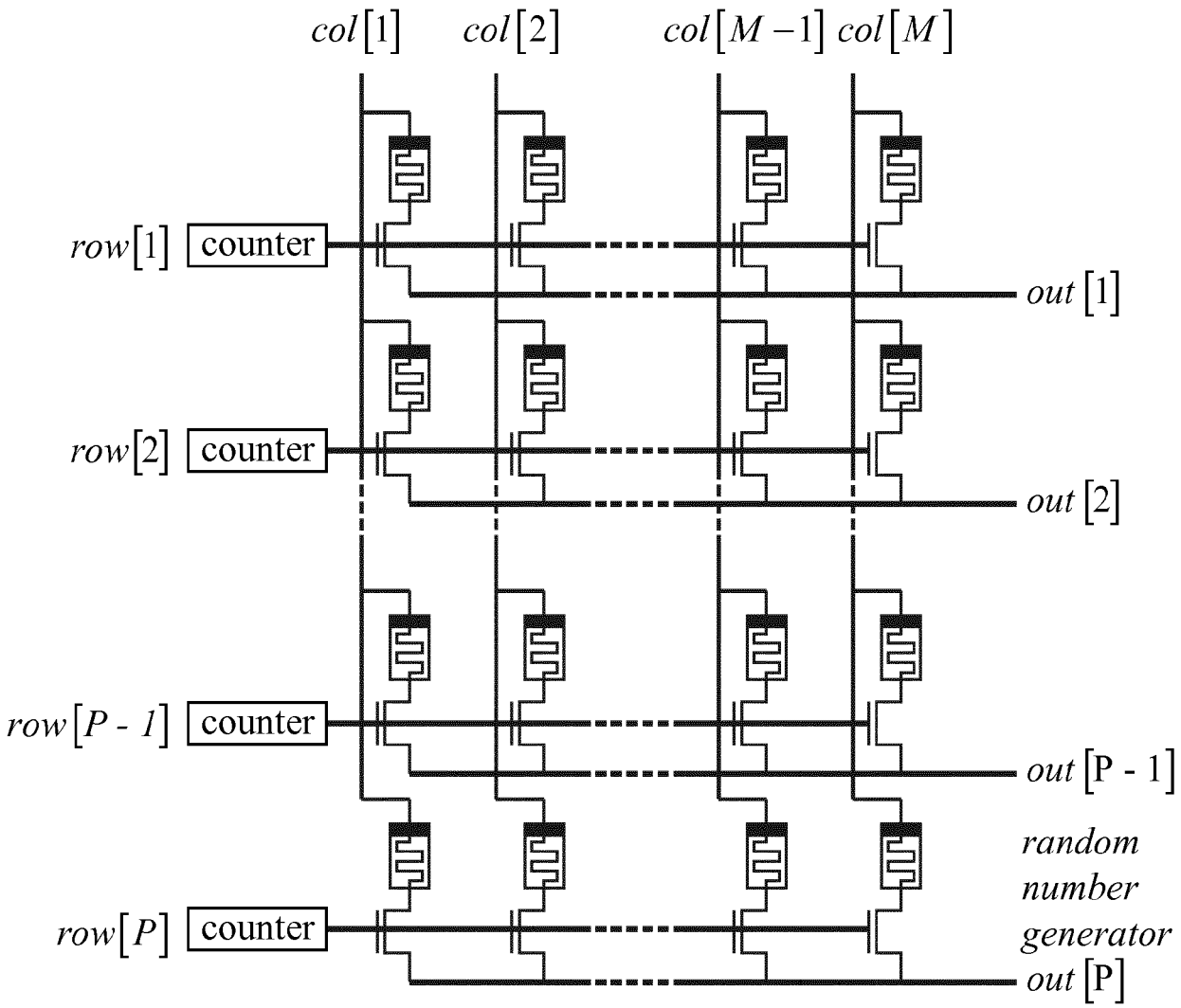
col[1]
col[2]
col[M −1]
col[M]
row[1]
counter
row[2]
counter
row[P - 1]
counter
row[P]
counter
out [1]
out [2]
out [P - 1]
random number generator
out [P]

Fig. 8

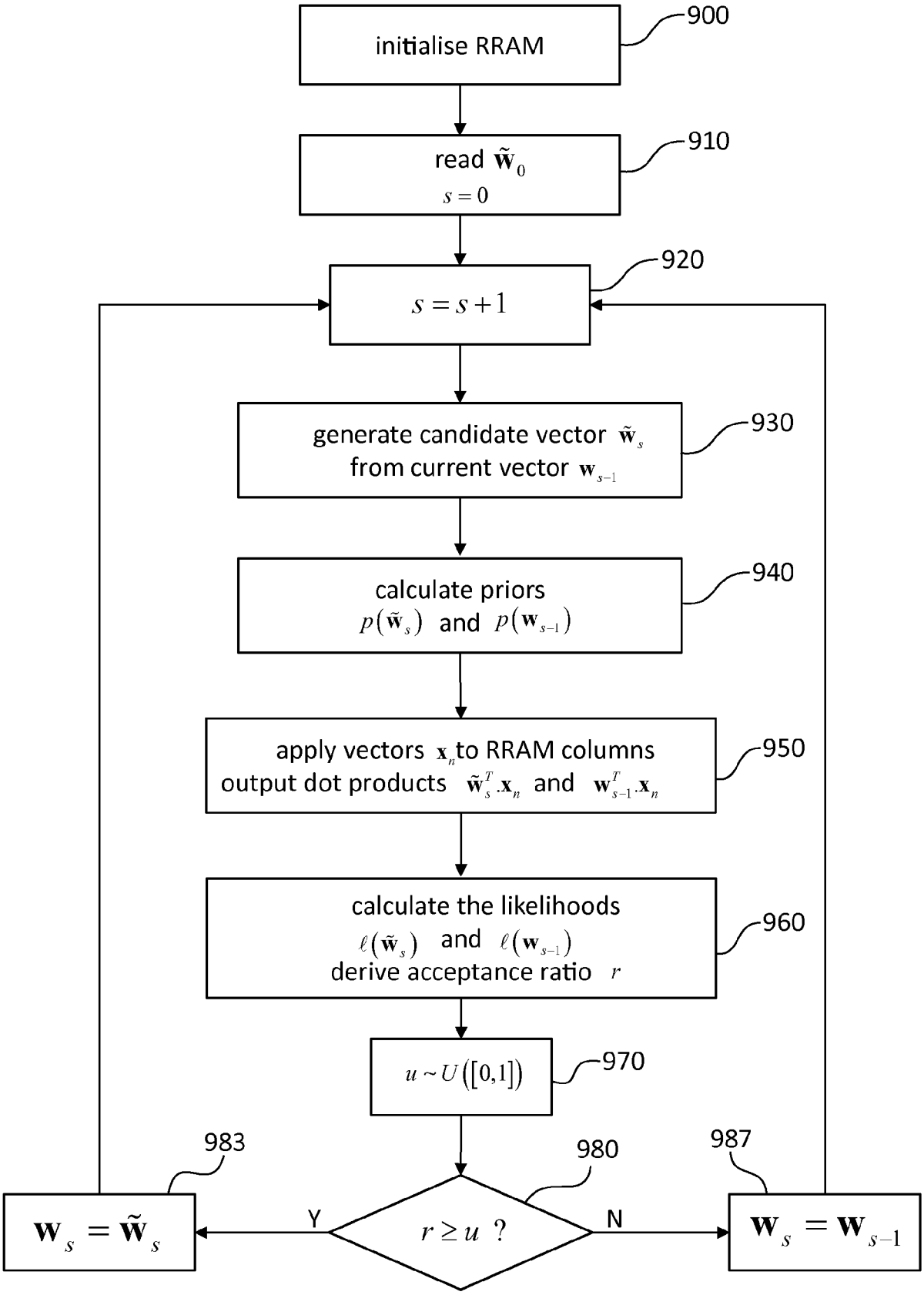
initialise RRAM
900
read $\tilde{\mathbf{w}}_0$
$s=0$
910
$s=s+1$
920
generate candidate vector $\tilde{\mathbf{w}}_s$
from current vector $\mathbf{w}_{s-1}$
930
calculate priors
$p(\tilde{\mathbf{w}}_s)$ and $p(\mathbf{w}_{s-1})$
940
apply vectors $\mathbf{x}_n$ to RRAM columns
output dot products $\tilde{\mathbf{w}}_s^T.\mathbf{x}_n$ and $\mathbf{w}_{s-1}^T.\mathbf{x}_n$
950
calculate the likelihoods
$\ell(\tilde{\mathbf{w}}_s)$ and $\ell(\mathbf{w}_{s-1})$
derive acceptance ratio r
960
$u \sim U([0,1])$
970
983
980
987
$\mathbf{w}_s = \tilde{\mathbf{w}}_s$
Y
$r \geq u$?
N
$\mathbf{w}_s = \mathbf{w}_{s-1}$

Fig. 9

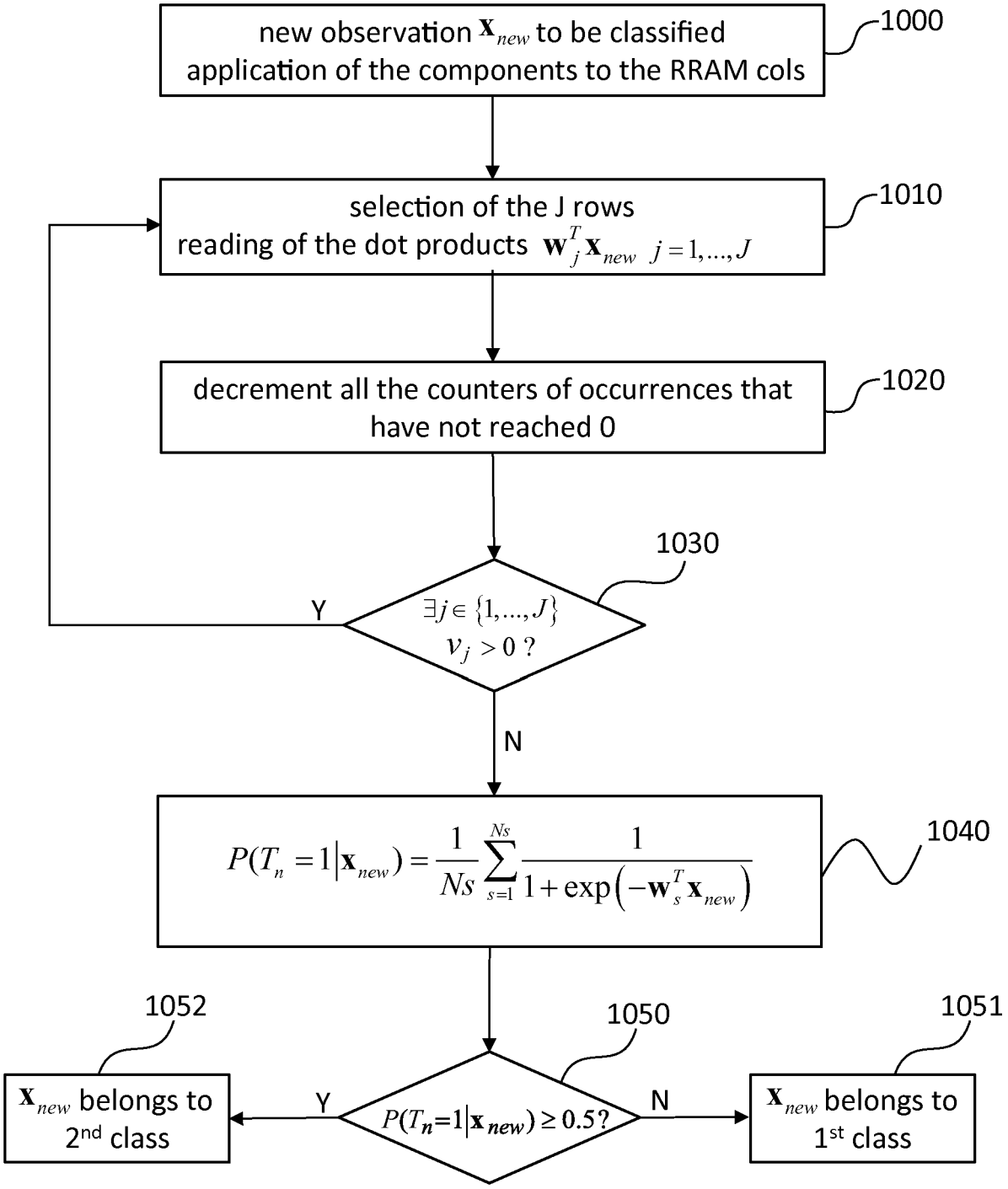
new observation $\mathbf{x}_{new}$ to be classified
application of the components to the RRAM cols
1000
selection of the J rows
reading of the dot products $\mathbf{w}_j^T \mathbf{x}_{new} \quad j = 1,...,J$
1010
decrement all the counters of occurrences that have not reached 0
1020
1030
$\exists j \in \{1,...,J\}$
$v_j > 0$?
Y
N
$P(T_n = 1 | \mathbf{x}_{new}) = \frac{1}{Ns} \sum_{s=1}^{Ns} \frac{1}{1 + \exp(-\mathbf{w}_s^T \mathbf{x}_{new})}$
1040
1052
1050
1051
$\mathbf{x}_{new}$ belongs to 2nd class
Y
$P(T_n = 1 | \mathbf{x}_{new}) \geq 0.5$?
N
$\mathbf{x}_{new}$ belongs to 1st class

Fig. 10

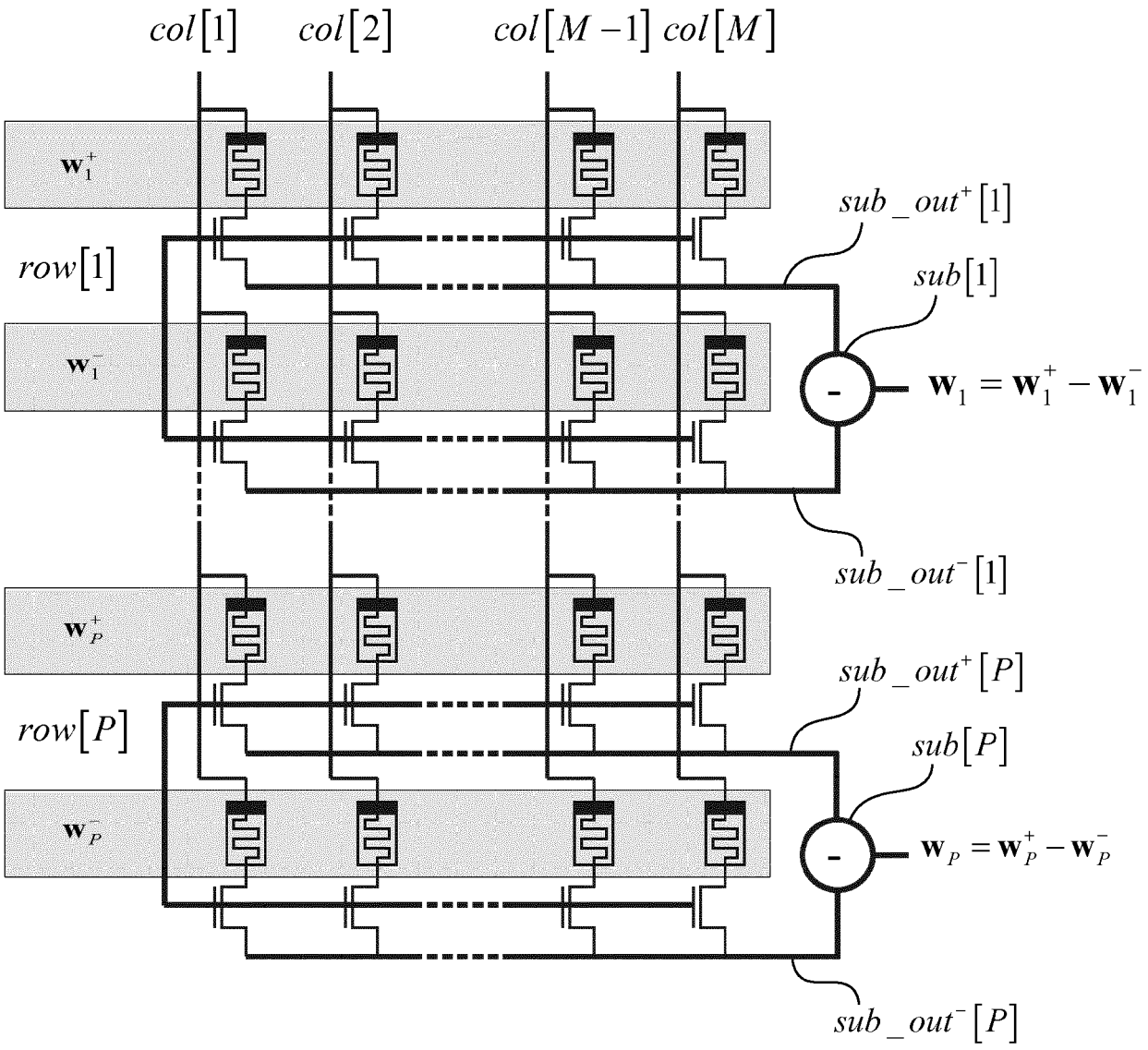
col[1]
col[2]
col[M −1]
col[M]
$\mathbf{w}_1^+$
$\mathbf{w}_1^-$
row[1]
sub_out+[1]
sub[1]
-
$\mathbf{w}_1 = \mathbf{w}_1^+ - \mathbf{w}_1^-$
sub_out−[1]
$\mathbf{w}_P^+$
$\mathbf{w}_P^-$
row[P]
sub_out+[P]
sub[P]
-
$\mathbf{w}_P = \mathbf{w}_P^+ - \mathbf{w}_P^-$
sub_out−[P]

Fig. 11

Europäisches Patentamt
European Patent Office
Office européen des brevets

EUROPEAN SEARCH REPORT

Application Number
EP 19 21 0420

DOCUMENTS CONSIDERED TO BE RELEVANT

Category	Citation of document with indication, where appropriate, of relevant passages	Relevant to claim	CLASSIFICATION OF THE APPLICATION (IPC)
Y	GROENEWALD P C N ET AL: "Bayesian computation for logistic regression", COMPUTATIONAL STATISTICS AND DATA ANALYSIS, NORTH-HOLLAND, AMSTERDAM, NL, vol. 48, no. 4, 1 April 2005 (2005-04-01), pages 857-868, XP027662479, ISSN: 0167-9473 [retrieved on 2005-04-01]	1,5-9	INV. G06N7/00
A	* page 858 - page 859 *	2-4	

Y	Akul Malhotra ET AL: "Exploiting Oxide Based Resistive RAM Variability for Probabilistic AI Hardware Design", , 16 November 2019 (2019-11-16), XP055693207, Retrieved from the Internet: URL:https://arxiv.org/pdf/1911.08555v1.pdf [retrieved on 2020-05-08]	1,5-9	
A	* page 1 - page 3 *	2-4	
	-----		TECHNICAL FIELDS SEARCHED (IPC)
A	AAYUSH ANKIT ET AL: "PUMA", ARCHITECTURAL SUPPORT FOR PROGRAMMING LANGUAGES AND OPERATING SYSTEMS, ACM, 2 PENN PLAZA, SUITE 701NEW YORKNY10121-0701USA, 4 April 2019 (2019-04-04), pages 715-731, XP058433490, DOI: 10.1145/3297858.3304049 ISBN: 978-1-4503-6240-5 * page 715 * * page 718 *	1-9	G06N

The present search report has been drawn up for all claims

Place of search	Date of completion of the search	Examiner
The Hague	12 May 2020	Bohn, Patrice

CATEGORY OF CITED DOCUMENTS

X : particularly relevant if taken alone
Y : particularly relevant if combined with another document of the same category
A : technological background
O : non-written disclosure
P : intermediate document

T : theory or principle underlying the invention
E : earlier patent document, but published on, or after the filing date
D : document cited in the application
L : document cited for other reasons

& : member of the same patent family, corresponding document

EPO FORM 1503 03.82 (P04C01)

REFERENCES CITED IN THE DESCRIPTION

This list of references cited by the applicant is for the reader's convenience only. It does not form part of the European patent document. Even though great care has been taken in compiling the references, errors or omissions cannot be excluded and the EPO disclaims all liability in this regard.

Non-patent literature cited in the description

- **SIMON ROGERS ; MARK GIROLAMI.** First course in Machine Learning. Chapman & Hall/ CRC Press, 2017 **[0014]**
- **C. ANDRIEU et al.** An introduction to MCMC for machine learning. *Machine Learning,* 2003, vol. 5, 5-43 **[0014]**
- **R. CARBONI ; D. LELMINI.** Stochastic memory devices for security and computing. *Adv. Electron. Mat.,* 2019, vol. 5 (1-27), 1900198 **[0036]**